Algorithm Spring made by Jurijus Pacalovas compression by calculus and reverse: Size of file should bigger than 1970 bytes before when want you to compress. if lenf1<2000:

Size is blocked 1970 bytes. bnk=bnk*255 ghjd=ghj*bnk cvz=cvz+ghdj

bnk=bnk*256 ghjd=ghj*bnk cvz=cvz+ghdj

Make smaller than when lenfg> 0: bits and save size 14311 bits when lenfg==0: save size 14320. When size 14320 bits take *111...**0X move to the end* **111...**0 or **111...**1, change last 1 to 0 put to the front of szx **0**00…1 and if long is 1 bit change last one to 0 or left 1. Make them together from right to left. Cout this size and save it in bytes than the count of this long and again of this long that should be 1 byte. When sizing 14310 bits save as 1111...0 and size will become 14311 bits change first 1 to 0.

Left 0x and 1111....0x

Save how many times was compressed by one byte.

check if lenfg==0: or if lenfg>0: lenfg mean when the size of data does not exist information on the block and save information about the first information not exist and change 255 to information not exist. if lenf1<=sssssw or sssssw<=2000 or qqqwz==2**30: check the size of the file and check 2**30: and save this file.